Real-Time Active Noise Cancellation with Simulink and Data Acquisition Toolbox

Vaibhav Narula[1], Mukul Sagar[2], Pranab Joshi[2],Puneet S. Mehta[2],Sudhanshu Tripathi[2]
[1] Amity School of Engineering and Technology(Electronics and Communication), New Delhi, India
Email: vaibhav3589@gmail.com
[2] Amity School of Engineering and Technology(Electronics and Communication), New Delhi, India
Email: {mukulsagar1989, joshi.pranab04, puneetmehta90, sudhanshutripathi14}@gmail.com

***Abstract*— This paper presents the feasibility of implementing single channel negative feedback Active Noise Cancellation technique using adaptive filters in Real-time environment[1]. In order to establish the suitability and credibility of LMS Algorithm for adaptive filtering in real world scenario, its efficiency was tested beyond system based ideal simulations. Within the MATLAB® software environment two different methods were used to perform Real-time ANC namely Simulink® and Data Acquisition Toolbox™. Human voice is used as test signal. For processing and performing adaptive filtering, Block LMS Filter was utilised in Simulink and Error Normalised Step Size algorithm was used in between input and output of Signals by DAQ (Data Acquisition) toolbox interface. A general method of using DAQ commands has been employed which also allows for almost any kind of complex real-time audio processing and is quite easy to follow.**

***Index Terms*— Active Noise Cancellation, Adaptive Filters, LMS, ENSSLMS, Simulink, Data Acquisition, Real-time**

I. Introduction

Active noise cancellation is basically the electro-acoustic generation of a sound field to cancel an unwanted existing sound field. The active noise cancellation system is known as an "adaptive" system as it can adapt itself to changing characteristics of the noise to be cancelled and changing environmental conditions that affect the acoustic field. An Active noise cancellation is basically the electro-acoustic generation of a sound field to cancel an unwanted existing sound field. The active noise cancellation system is known as an "adaptive" system as it can adapt itself to changing characteristics of the noise to be cancelled and changing environmental conditions that affect the acoustic field. An Adaptive algorithm used to optimize the FIR or IIR filter weights "on-line" in active noise and vibration control systems are essentially derivations of the adaptive algorithms used in systems such as telephone echo-cancellers and adaptive optics in telescopes to cancel unwanted optical "noise" and thus enhance signals from distant stars. One such algorithm is Least Mean Square algorithm (LMS). The LMS (least mean squares) algorithm is a member of stochastic gradient algorithms and approximation of the steepest descent algorithm which uses an instantaneous estimate of the gradient vector of a cost function. The estimate of the gradient is based on sample values of the tap-input vector and an error signal. The initial idea is to use the LMS (Least-Mean-Square) algorithm to develop an adaptive filter that can be used in ANC (Active Noise Cancellation) applications. The method uses a noisy signal as primary input and a reference input that consists of noise correlated in some arbitrary manner with the primary noise. By adaptively filtering and subtracting the reference input from the primary input, the output of the adaptive filter will be the error signal, which acts as a feedback to the adaptive filter. With this setup, the adaptive filter will be able to cancel the noise and obtain a less noisy signal estimate. Adaptive filter is generally defined as a filter whose characteristics can be modified to achieve some objective and is usually assumed to accomplish this adaptation automatically, without the need for substantial intervention by user. Here, an adaptive filter is needed that is capable of altering its impulse response as required for the noise cancellation application. Figure 1 shows the general structure of an adaptive filter. Figure 1 is composed of three blocks as shown. The first block is Adaptive filter block which shows what type of digital filter is used in the design of the adaptive filter. The second block depicts filter adaptation rules which show the updating algorithm used for the adaptive filter to update its parameters, and last block as its name shows performs the quality assessment or simply comparison in an adaptive system, the output of this block is the error signal which is fed back to filter adaptation rules block to find new parameters. In Figure 1, the filter could be implemented in analogue or digital, which here a digital filter is used.

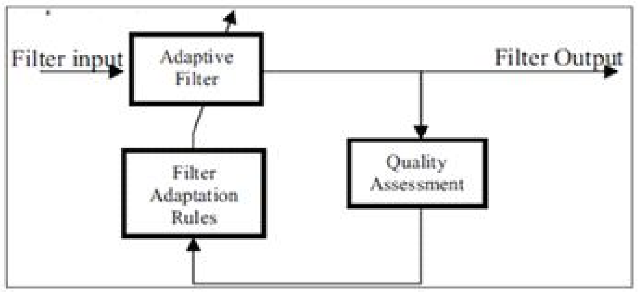

Figure 1. The general structure of an Adaptive Filter

A. Related Work

From the retrospective consideration it can be said that scads of work have been done in the domain of Realtime ANC. Till now live audio data(input) corrupted by broadband noise have not yet been treated for filtration through adaptive modus in Simulink environment or using Data Acquisition Toolbox. But almost every time [see [5],[6],[7]] live audio processing (be it noise cancellation or some other) has been attempted and successfully performed using Embedded DSP environment only. Simulink and DAQ toolbox have never been directly used in true sense for handling live streaming audio input in context of Active Noise cancellation through adaptive filters. The novel approach followed in this newspaper is very simple, direct and specially focussed to enable even a novice to accomplish DSP chores of certain complexity level with live realtime audio using Simulink as well as Data Acquisition toolbox and basic MATLAB commands. This method requires only winsound audio adapter(present in all multimedia computers) for audio acquisition unlike other data Acquisition techniques which may require expensive sophisticated sound adapters.

II. REAL-TIME DIGITAL SIGNAL PROCESSING

In realtime DSP, the real world (analog) data is input through input transducers , conditioned and converted to digital data via an ADC. The Digital Processor processes it and DAC converts output of processor to analog data which is then conditioned through reconstruction filter and finally supplied out of the system via output actuators. One mandatory condition of the Real-Time DSP system is that the processing time must be less than the data transmission time. Real-time signal processing is generally done for animated and continuous data such as audio, video and data streams where as batch processing (even though also employed for stored audio and video data) is used for static data clusters like image, files etc. There are three structural levels of real-time processing briefed as follows

- Stream processing : all computations with one input sample are completed before the next input sample arrives
- Block processing : each input sample is stored in memory before any processing occurs upon it. After a bunch of input samples have arrived, the entire collection of samples is processed at once.
- Vector processing : systems with several input and/ or output signals being computed at once: can work with streams or blocks

Further in this paper scope and purpose of Active Noise Cancellation technique using Adaptive Filtering has been experimented and explored in context of Simulink and DAQ Toolbox interface.

A. Implementation Using Simulink

Simulink, [3]developed by MathWorks, is a commercial tool for modelling, simulating and analyzing multidomain dynamic systems. Its primary interface is a graphical block diagramming tool and a customizable set of block libraries. It offers tight integration with the rest of the MATLAB environment and can either drive MATLAB or be scripted from it. Simulink is widely used in control theory and digital signal processing for multidomain simulation and Model-Based Design. After defining a model one can integrate it using a choice of integration methods either from Simulink menus or by entering commands in MATLAB's command window. Simulations can be of the realtime or non realtime in nature of interaction with the user. Stateflow is another entity which is a part of Simulink environment. The Stateflow block is a masked Simulink model. Stateflow builds an S-function that corresponds to each Stateflow machine. This S-function is the agent Simulink interacts with for simulation and analysis. Due to complex behaviour of S-function, any realtime simulation experiences a bit of latency in its performance. For true to form Realtime simulations MATLAB has developed an equipment to Simulink which is called Real-Time Workshop(now Simulink Coder). The Real-Time Workshop product generates and executes stand-alone C/ C++ codes for developing and testing algorithms modelled in Simulink and Embedded MATLAB code. The resulting code is simpler than S functions and can be used for many real-time and non-real-time applications, including simulation acceleration, rapid prototyping, and hardware-in-the-loop testing. You can tune and monitor the generated code using Simulink blocks and built-in analysis capabilities, or run and interact with the code outside the MATLAB and Simulink environment.

But getting a hand with Realtime Workshop and actually working on it is tough for the beginners in MATLAB and Simulink society. So simple Simulink Models with realtime I/ O and processing capabilities can be efficient to begin with especially for non advanced users such as students.

1) Simulink Model For Noise Cancellation

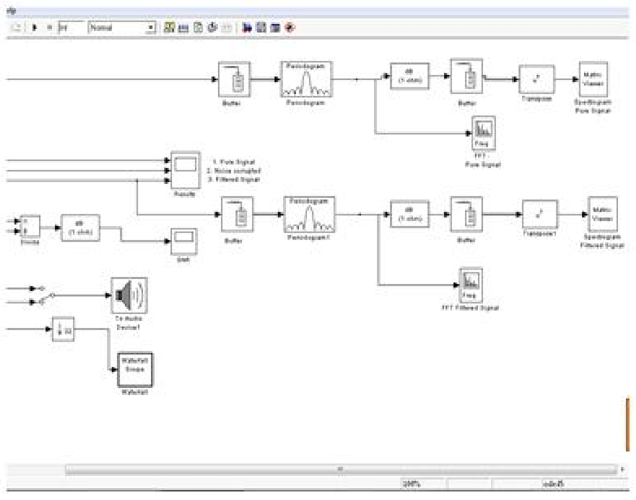

Figure 2. Simulink Module for FFT and Spectrogram output

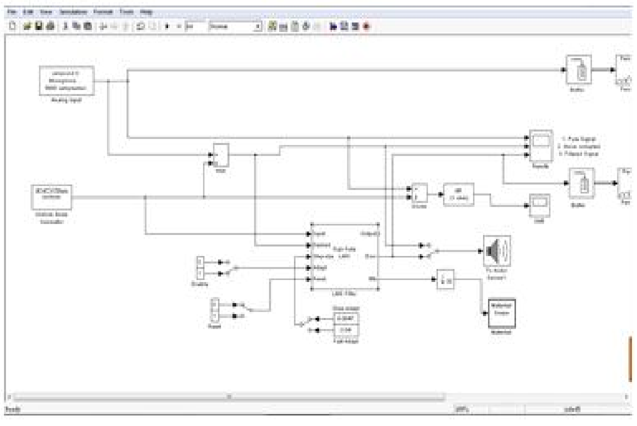

Figure 3. Acoustic Noise Canceller

2) Simulink Model Description

The Simulink Model illustrated here consists of two modules . Figure 2 module uses the Analog Input block to acquire live signals from a data acquisition device(‘winsound’ adapter and microphone) into Simulink. The sound card is selected as the input device in the Analog Input block. The Simulink model performs spectral estimation through periodogram method. A periodogram computes the spectral content of a signal over time. The input signal is a real-time audio through a microphone which is buffered into frames, with 128 samples per frame. Each frame is then windowed using a Hamming window function, and this blockset computes the FFT for the windowed frame. The system collects the FFTs for successive frames and plots them to produce a spectrogram. Even though a Windows sound card was used for this demo, this model can be easily updated to connect models to other supported data acquisition devices. This provides the flexibility to use the same Simulink model with different data acquisition device.

Figure 3 module is the Acoustic Noise Canceller. This module uses the acquired Signal from Analog Input block i.e. data acquisition device in a real-time manner. Running FFT of this signal is displayed using above module. Then this signal is summed up with white noise from uniform noise generator and fed to the Input port of LMS Adaptive Filter block. White noise directly from noise generator is plugged into the desired Signal port of the LMS Filter Block. Filter length of LMS block is 60 and Sign-Data variant of LMS Algorithm is used for adaptive processing. Step size of LMS Adaptive filter is switched between higher constant value for fast but less accurate adaptation and lower constant value for slower but more precise performance through manual switch. The difference of desired Signal and Input Signal is the error Signal which is obtained from error Signal output port of LMS block. The Signal from this port is the Filtered Signal from which Noise has been adaptively removed out. Signal from output port is the Noise corrupted Signal. Sample based signals from these two output ports are supplied to the Analog Output Block which delivered the acquired samples to Audio Output device of the System in frame based manner. Single Output Device block is used and signals from output ports of LMS blocks are switched manually. Output from weights port is down sampled 32 times and adaptation in the weights is viewed from Waterfall graph in time bound manner. Simulation only works fine in normal mode with simulation stop time set to Infinity (optional).

3) Scope Output

Figure 4 is the scope output of Simulink Model. In this scope the top axes depicts pure audio signal entering the system. Second axes shows the resultant signal in voltage coordinates after getting corrupted by Uniform Noise thus generated. The bottom axes is the signal obtained after Adaptive filtering.

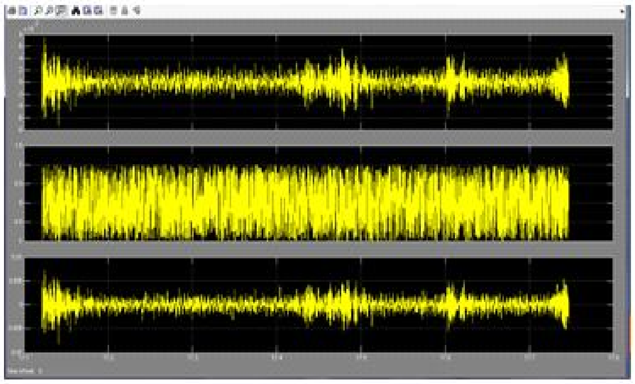

Figure 4. Input Signal, Noise Corrupted Signal & Filtered Signal

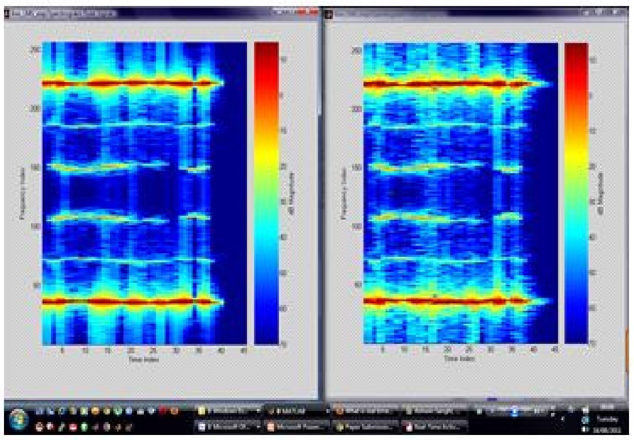

Figure 5. Spectrogram of Input(left) and Filtered(right) Signal

B. Implementation Using Data Acquisition Toolbox[4]

Data Acquisition Toolbox™ provides a complete set of tools for analog input, analog output, and digital I/O from a variety of PC-compatible data acquisition hardware. The toolbox lets you configure your external hardware devices, read data into MATLAB® and Simulink® for immediate analysis, and send out data.

1) DAQ Procedure

First of all create(declare) analog input object and figure. Then using ‘set’ command update userdata of figure and analog input. Now create(define) the analog input object and get first Adaptive output in the subsequent function. Establish a field in the object to collect batch/block of data samples . The length of this block/batch can be optimized (for performance) by setting proper values of ‘SampleRate’, ’SamplesperTrigger’, ’Triggertype’, ’TriggerRepeat’ and ‘TimerPeriod’ properties of the Data Acquisition object(ai) in accordance with each other and input signal.

Create a function for display figures and plots. After storing handles in the data matrix configure callback to update the display. In the callback function any audio processing algorithm can be written. This algorithm will be applied to every block/batch of data samples acquired and stored in the internal buffer following every trigger. Output of the algorithm will update the previous output after every fixed interval(set on ‘TimerPeriod’ property of ‘ai’ object). This refresh period if carefully selected by managing trade-off between expected input device latency and processor speed, a live realtime audio processing can be performed and thus visualised using line plots. The last step to be carried in the callback function is to link the plots with triggered input sequence of samples and also with the data batch/block output from algorithm. Error Normalised Step Size LMS[2] algorithms is used for adaptive filtering in the DAQ method. For the purpose of reference, demoai_fft.m was used to understand and implement the procedure hereby mentioned. This file can be found in Data Acquisition Demos in MATLAB help having label of " FFT Display of an Incoming Signal".

2) Error Normalised Step Size LMS(ENSSLMS) Algorithm[2]

In the ENSS proposed algorithm, the variable step is inversely proportional to the squared norm of the error vector. The length of the error vector is equal to the instantaneous number of iterations. Thus, the weight update equation of the ENSS becomes

$$w(n+1) = w(n) + \frac{\mu}{1+\mu\|e_n\|^2} e(n)Ni(n) \qquad (1)$$

where

$$\|e_n\|^2 = \sum_{i=0}^{n-1} e^2(n-i) \qquad (2)$$

is the squared norm of the error e(n), estimated over its entire updated length and μ is the step size.

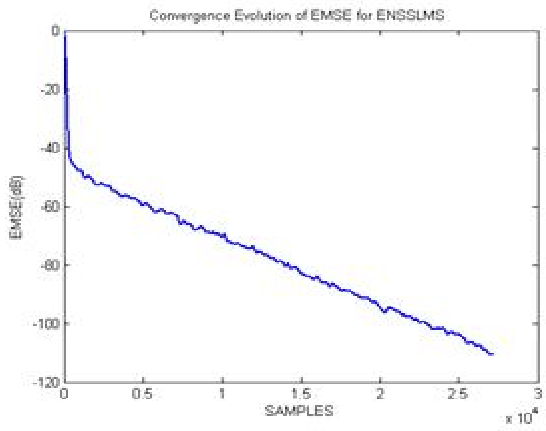

Figure 6. Evolution of EMSE(Excess Mean Square Error) for ENSSLMS with identical set of inputs

The plot in Figure 6 shows the convergence rate of error signal (EMSE) when Noise Cancellation is carried out with Adaptive Filter , designed using ENSS Algorithm. The single block length processed during interval of every callback loop is of 4096 samples. From the convergence plot (similar set of signals were used in simulation as well as realtime experimentation) , its visible that convergence rate is 50-55 dB for initial 4000 samples which is quite acceptable for practical purposes(exercised in the paper).

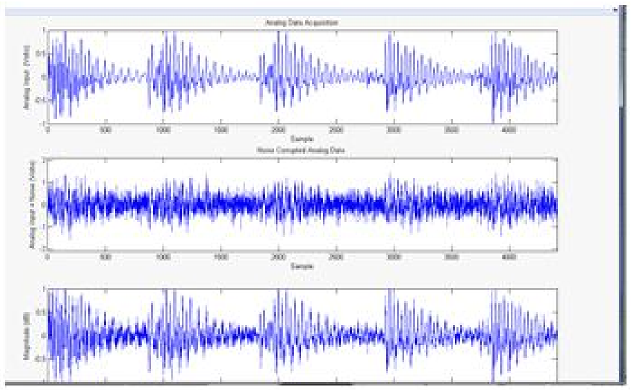

Figure 7. Input Signal, Noise Corrupted Signal & Filtered Signal

3) Figure Scope output

The above Figure 7 demonstrates the real-time plot of Noise Cancellation using Data acquisition toolbox commands and ENSS-LMS[2] adaptive filter . In this figure top axes shows the pure input signal acquired via acquisition device. The middle axes is the noise corrupted plot of above signal. Audio signal after adaptive filtering is shown in the bottom axes.

III. DISCUSSION OF OUTPUTS/RESULTS

In oscilloscope output from Simulink model one can see an elegant result. The noise is almost completely removed. The SNR for noise corrupted signal is 0-5 dB and post filtration SNR is approx 40 dB. Note the magnification of Y axis for middle plot is 1000x compared to top and bottom plots. Error convergence rate for every 4000 samples is 50-55 dB is also clear from Filtered Signal output of Figure 7 in left to right decrease of noise (disturbance) magnitude. Random noise of fixed variance (0.1) was generated in an internal function and summed up with the incoming system acquired audio. Average SNR calculated was 20dB. Using tic-toc function, time elapsed for entire callback function was found to be 0.01 to 0.015 seconds. So the plot update time can be adjusted form anything to minimum of 0.02 seconds. Note that the length of y axis of middle plot is twice that of top and bottom plots.

IV. APPLICATION AND FUTURE SCOPE

This method of data acquisition and signal processing can be used for any kind of live audio stream and complex signal processing algorithms can be tested on them in Realtime manner. The Simulink method is much slower than DAQ method from realtime perspective. To improve this scene Simulink Coder (earlier called Real Time workshop) can be used which generates and executes C & C++ code from Simulink model. The generated code can be used for Realtime Applications with much more efficiency and much less latency.

CONCLUSIONS

This paper aimed at performing Realtime active nose cancellation using Adaptive Filtering Technique through Simulink Environment and DAQ toolbox commands. The aim was thus suitable fulfilled and basic level feasibility of Realtime Signal Processing using both the techniques was hereby demonstrated upto acceptable level of performance. The structural level of Realtime processing employed in this paper was block-vectored type.

REFERENCES

[1] Gaurav Saxena, Subramaniam Ganesan and Manohar Das, "Real-time Implementation of Adaptive Noise Cancellation", in IEEE International Conference on Electro/Information Technology, 2008. EIT 2008, pp. 431-436, 27 June 2008

[2] Z. Ramadan and A. Poularikas, "An adaptive noise canceler using error nonlinearities in the LMS adaptation," in Proc. IEEE Southeast Con 2004, Mar. 2004, vol. 1, pp. 359–364.

[3] Simulink, MATLAB version R2010a

[4] Data Acquisition Toolbox, MATLAB version R2010a.

[5] R. Martínez, A. Álvarez, V. Nieto, V. Rodellar and P. Gómez, "Implementation Of An Adaptive Noise Canceller On TMS320C 31-50 for Non Stationary Environments", in Proceedings on the 13th International Conference on Digital Signal Processing, Santorini, Greece, 2-4 July, 1997, pp. 49-52

[6] Cristina Gabriela Saracin, Marin Saracin, Mihai Dascalu, Ana-Maria Lepar, "Echo Cancellation Using The LMS Algorithm", in U.P.B. Sci. Bull., Series C, Vol. 71, Iss. 4, 2009, pp. 167-174

[7] Bambang Riyanto, "Real-time DSP Implementation of Active Noise Control for Broadband Noise Using Adaptive LMS Filter Algorithm", in Proceedings of the International Conference on Electrical Engineering and Informatics Institut Teknologi Bandung, Indonesia June 17-19, 2007, pp. 718-722